Universidade de Brasília **27/04/2017**
Instituto de Ciências Exatas **Profa Priscila Solis**
Departamento de Ciência da Computação **Eduardo Scartezini 14/0137084**
Teleinformática e Redes 1 **Lucas Santos 14/0151010**

Projeto Captura de Pacotes

1 Endereço IP da aplicação no computador e endereço IP da máquina remota.

Computador: 172.31.222.216

Máquina remota: 212.227.179.107

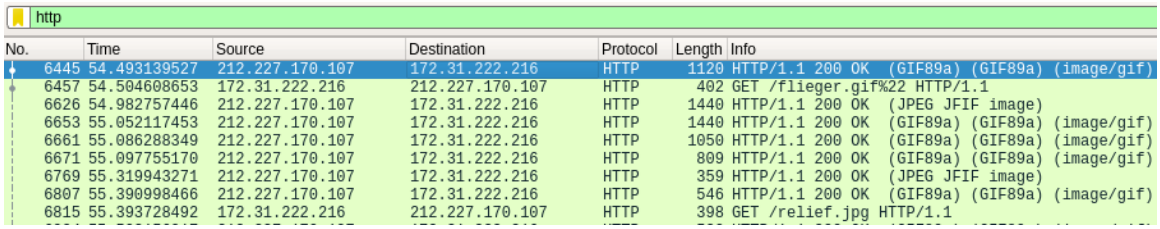

Figura 1: O campo *Source* do pacote destacado representa o IP da máquina remota e o campo *Destination*, o IP da aplicação no computador. OK é enviado pelo servidor(máquina remota), GET é enviado pelo cliente(aplicação no computador).

2 Três protocolos que foram observados nos pacotes capturados e as funções dos mesmos.

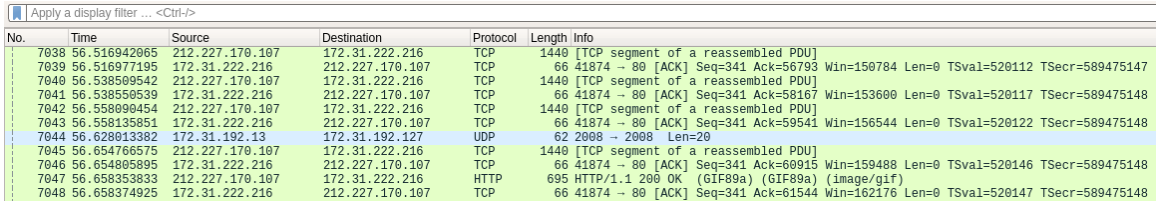

Figura 2: Na figura acima, podem se perceber os tipos de protocolo TCP, UDP E HTTP.

Protocolo TCP (Protocolo de Controle de Transmissão), protocolo implementado na camada de transporte a qual é responsável por fornecer comunicação logica entre dois hospedeiros, o TCP faz isso de forma orientada à conexão, sendo um protocolo que garante uma entrega confinável dos pacotes, alem disto o TCP implementa um controle de congestionamento um uso melhor e mais justo da rede.

Protocolo UDP (Protocolo de Datagrama de Usuário), também um protocolo da camada de transporte, porém o UDP implementa as responsabilidades desta camada de uma forma diferente, o UDP não é orientado a conexão e muito menos tem um controle de congestionamento, não garantindo uma entrega confiável, mas em compensação é um protocolo muito mais simples e com um *overhead* menor.

Protocolo HTTP (Protocolo de Transferência de Hipertexto),o protocolo da camada de aplicação, responsável pela troca de mensagens entre dois sistemas finais, mais comumente utilizado em paginas web.

3 Esquema de encapsulamento até a camada de enlace de uma sequência de pacotes capturados.

Vemos nas figuras a seguir o encapsulamento de três pacotes diferentes, o primeiro pacote usa um protocolo da camada de aplicação o HTTP, o TCP na camada de transporte, o IP na camada de rede, o Ethernet II na camada de enlace e tudo isso é transmitido em uma camada física. Já no segundo pacote é usado o UDP na camada de transporte, e por fim o ultimo pacote usa TCP na camada de transporte e não possui camada de aplicação. É interessante notar que por mais que em uma camada o protocolo utilizado seja diferente as outras camadas continuam oferecendo o mesmo serviço, mostrando a importância e a modularidade de uma arquitetura em camadas.

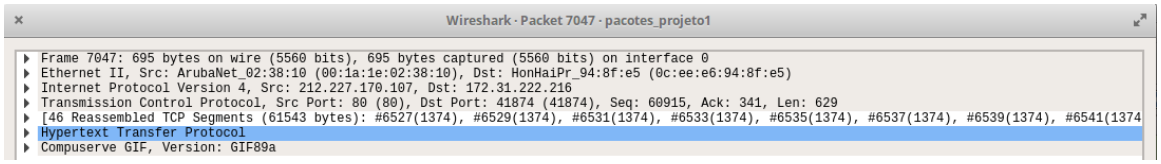

Figura 3: Esquema de encapsulamento do protocolo HTTP, contendo as camadas de: aplicação, transporte, rede, enlace e física

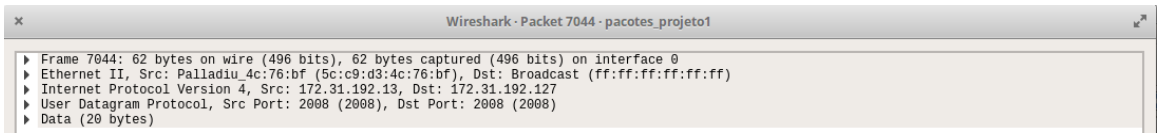

Figura 4: Esquema de encapsulamento do protocolo UDP, contendo as camadas de:

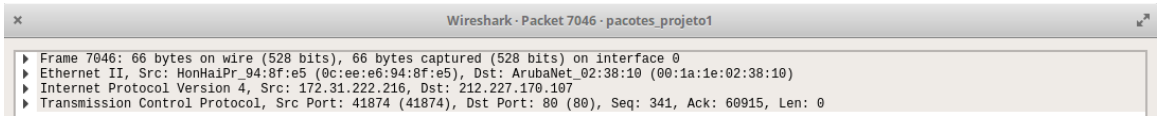

Figura 5: Esquema de encapsulamento do protocolo TCP, contendo as camadas de: transporte, rede, enlace e fisica

4 Formato do quadro IP, especificando e detalhando cada um dos conteúdos do cabeçalho.

- **Version**:
  - Valor: 0100 = 4;
  - 4 bits que especificam a versão do protocolo IP. A partir do número da versão, o roteador pode examinar o datagram IP. Foi capturado o datagrama com *Internet Protocol Version* 4 (IPv4).
- **Header Length**:
  - Valor: 0101 = 5 * 32 bits = 160 bits = 20 bytes.
  - Como um datagrama IPv4 pode conter número varável de opções incluidas no cabeçalho, estes 4 bits, que é o número de palavras de 32 bits, utilizados para determinar onde os dados começam. Foi capturado um datagrama com o valor do campo igual a 5 que indica o tamnho de 20 bytes.

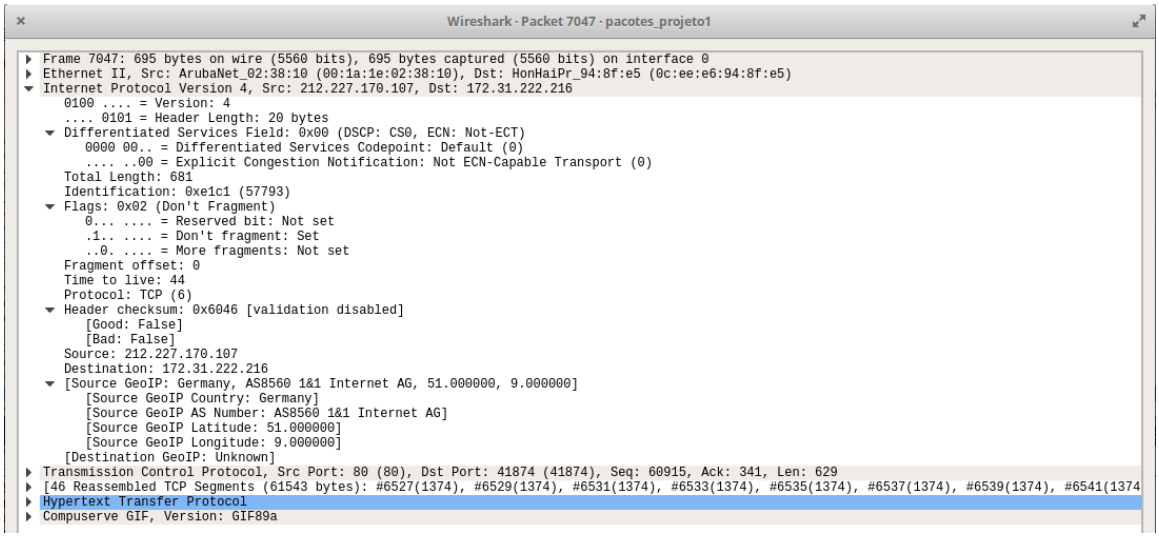

Figura 6: Formato do quadro IP do protocolo HTTP.

- **Differentiated Services Length**:
  - Valor: 0x00, DSCP: Default, ECN: Not ECN-Capable Transport;
  - Os bits de tipo de serviço foram incluídos no IPv4 para diferenciar o tipo de datagrama IP, por exemplo, que requerem baixo atraso, alta vazão ou confiabilidade. Pode ser útil para distinguir datagramas de tempo real como usados em VoIP.
- **Total Length**:
  - Valor: 681;
  - Comprimento total do datagrama IP (Cabeçalho e dados) medidos em bytes.
- **Identification**:
  - Valor: 0xe1c1, Reserved Bit: Not set, Don't fragment: Set, More fragments: Not set;
  - Identificação de um fragmento do datagrama.
- **Flags**:
  - Valor: 0x02;
  - Controle da fragmentação do datagrama.
- **Fragment Offset**:
  - Valor: 0;
  - Valor de deslocamento do fragmento.
- **Time To Live**:
  - Valor: 44;
  - Campo de tempo de vida de um datagrama, serve para garantir que o datagrama não fique circulando para sempre na rede. Esse campo é decrementado em uma unidade cada vez que o datagrama é processado por um roteador. Se TTL chegar a 0, o datagrama deve ser descartado.
- **Protocol**:
  - Valor: 6 (TCP);
  - O valor do campo indica o protocolo da camada de transporte específico ao qual a porção de dados desse datagrama IP deverá ser passada. O número de protocolo é o elo entre as camadas de rede e transporte.

- **Header checksum**:
  - Valor: 0x6046;
  - Serve para identificar datagramas recebidos com erros. É calculado tratando cada 2 bytes do cabeçalho como se fossem um número e somando estes números usando complementos aritméticos de 1. Um roteador calcula o valor do *checksum* para cada datagrama IP recebido, se um erro for detectado, em geral, o datagrama é descartado.
- **Source**:
  - Valor: 212.227.170.107;
  - IP de origem do pacote.
- **Destination**:
  - Valor: 172.31.222.216;
  - IP de destino do pacote.
- **Source GeoIP**:
  - Valor: Germany, AS8510 1&1 Internet AG, 51, 9;
  - Informação geográfica do IP de origem do pacote.
- **Destination GeoIP**:
  - Valor: Unknown;
  - Informação geográfica do IP de destino do pacote.

5 O formato do cabeçalho do pacote Ethernet, na camada de enlace, descrevendo cada um dos campos e detalhando como funciona o controle de erro nesta camada a partir do conteúdo do campo CRC.

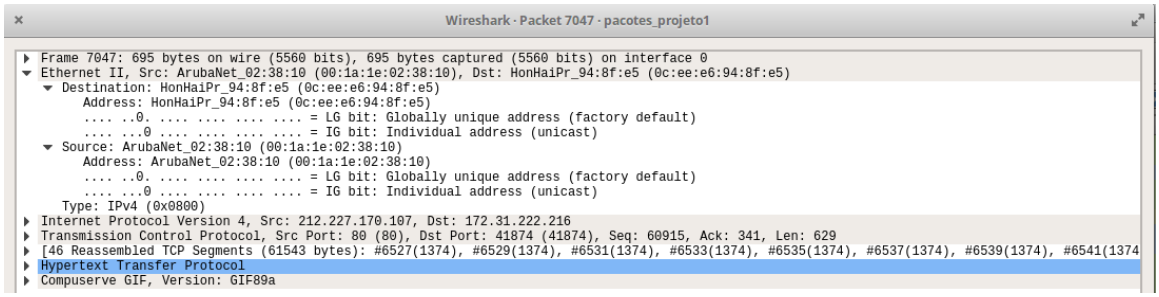

Figura 7: Formato do cabeçalho do pacote Ethernet.

- **Destination**:
  - Valor: HonHaipR_94:8f:e5;
  - Endereço MAC de destino.
- **Source**:
  - Valor: ArubaNet_02:38:10;
  - Endereço MAC de origem.

- **Type**:
  - Valor: 0x0800 = IPv4;
  - Protocolo utilizado na camada de rede.
- **Data**:
  - Valor:
  - Dados.
- **FCS**:
  - Valor:
  - Campo de verificação da corretude da mensagem.

5.1 Cálculo do campo CRC.

O cálculo do campo CRC do pacote examinado é feito por meio de um polinômio gerador pré-definido $g(x) = 0x04c11db7$, chamado também de CRC-32. O conteúdo do pacote é deslocado de 26 bits para a esquerda (equivalente ao grau do polinômio gerador) gerando o *encoding* $e(x)$. A partir disto, desloca-se também o polinômio gerador $g(x)$ até o este ter o mesmo grau de $e(x)$, assim são calculados sucessivas operações **xor**, deslocando o $g(x)$ sempre para o maior grau do resultado obtido. Ao final têm-se 4 bytes que definem o campo CRC do pacote.

Um programa na linguagem C (**CRC32.c**) foi desenvolvido para o cálculo do CRC do pacote específico 6904 (*pacotes_projeto1.pcapng*) que utiliza o protocolo *Ethernet II* na camada de enlace. O CRC de um pacote que utiliza *Ethernet II* é feito por meio de um código polinomial, que possui como polinômio gerador o CRC32, como já dito anteriormente. Mais detalhes sobre o cálculo do CRC deste pacote estão disponíveis no executável e na implementação em si.